

Use of Feature Similarity for AHC Variants in Text Clustering

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—In this research, we propose the AHC variants which consider the feature similarities and apply them to the text clustering. The initial AHC algorithm was previously modified into the version which did so, as the approach to the task. In this research, we mention the three AHC variants, in the view of their traditional versions: one where the data clustering proceeds in the bottom-up direction with the similarity threshold, one where it allows any merge of more than two pairs, and one where clusters are merged based on their radius. The three AHC variants are modified into the versions which consider the feature similarities, as the text clustering tools, as well as the initial AHC algorithm. This research is aimed to improve the clustering performance by proposing the modified versions of the AHC variants.

I. INTRODUCTION

The text clustering is defined as the segmentation of a single text group into several groups, each of which contains content based similar texts. A text as a clustering target is decomposed into paragraphs, each paragraph is decomposed into sentences, and a sentence is decomposed into words. The process of text clustering is to define a similarity metric between texts and apply an unsupervised learning algorithm to the task. The task of this research is the hard text clustering where each text is arranged into only one cluster and will be expanded into the soft text clustering or the hierarchical text clustering in subsequent research. The process of managing texts is fully automated by combining the text clustering with the text categorization.

This research is motivated by the successful results from using the AHC algorithm which considers the feature similarities in the text clustering. The similarity between numerical vectors was computed by using the semantic dependencies among features, in order to solve the poor discriminations among sparse numerical vectors which represent texts. The AHC algorithm was modified by the similarity metric between two numerical vectors which considers both the similarities among features and the similarities between feature values, as the approach to the text clustering. The modified version of AHC algorithm was empirically validated as its better performance than the traditional version in clustering texts based on their contents. We modify the AHC variants into the version which considers the feature similarities, as well as the AHC algorithm and apply them to the text clustering.

Let us mention the three AHC variants as the approaches to the text clustering. In their first variant, clustering data items proceeds in the bottom-up direction with the similarity threshold, instead of a desirable number of clusters. In the second variant, for each iteration, multiple cluster pairs are allowed to be merged. In the third variant, clusters are merged, depending on a radius of a particular cluster. The three AHC variants are modified into the versions where the similarity which considers the feature similarities is used and applied to the text clustering.

Let us mention some benefits from this research. The poor discrimination among numerical vectors from their sparseness is solved by proposing the similarity metric between numerical vectors which considers the feature similarity. The clustering speed is improved by replacing the AHC algorithm by the AHC variants which allow more than one cluster to be merged. The clustering performance is improved by modifying the AHC variants into the feature similarity-based versions. We provide the more recommendable approaches for implementing the text clustering system, with respect to both the speed and the clustering performance.

Let us mention the organization of this paper. In Section II, we explore the previous works which are relevant to this study. In Section III, we describe in detail what is proposed in this research. In Section IV, we mention the significances of this research and the remaining tasks. This paper is composed of the four sections.

II. PREVIOUS WORKS

This section is concerned with the previous works which are relevant to this research. It is intended to expand the style of modifying the AHC algorithm to its three variants. The directions of exploring previous works are derivation of AHC variants, modification of the standard AHC algorithm, and the cases of using the feature similarities for improving machine learning algorithms. The distinguishable points of this research are derivation of three AHC variants from the standard version, modification of them with the feature similarities, and application of them to the text clustering. This article is intended to explore previous works for providing the background for this research.

Let us explore the previous cases of deriving AHC variants. In 2012, Murtagh and Contreras mentioned the

possibility of deriving various versions of AHC algorithm and a particular variant which uses nearest neighbors [2]. In 2013, Sasirekha and Baby presented various similarity metrics between vectors and strategies of defining a similarity between clusters [3]. In 2015, Nazari et al. proposed the AHC algorithm which merges two clusters based on their common nearest neighbors [4]. In the AHC algorithms which are mentioned in above previous literatures, only one pair of clusters is merged in each iteration.

Let us explore the previous works which are concerned with the application of the modified version of the standard AHC algorithm to text clustering. In 2018, Jo proposed initially that the modified version should be applied to the text clustering [5]. In 2018, Jo validated empirically the modified AHC algorithm in the text clustering [6]. In 2023, he prepares the journal article which describes the modified AHC algorithm and its application to the word clustering for its publication [10] as well as text clustering. What is provided by the previous works becomes the basis for this research.

Let us mention the previous works on the cases of using the feature similarities as the basis for modifying the KNN variants in this research. In 2002, feature similarities were used for selecting some features for using unsupervised learning algorithms by Mitra et al. [1]. In 2018, feature similarities were used for modifying the KNN algorithm as an approach to word classification by Jo [7]. In 2019, he used the modified version for classifying texts [9]. Feature similarities are used for solving the sparseness in numerical vectors.

Let us mention the differences of this research from the previous works which are explored above. In this research, we derive the AHC variants as fast clustering algorithms by allowing merging multiple cluster pairs for each iteration. The three AHC variants are modified into versions which consider the feature similarities and adopted for implementing the text clustering system. The feature similarities are used for modifying the AHC variants as the approaches to the word clustering. We will describe the modified AHC variants in Section III.

III. PROPOSED WORKS

This section is concerned with what is proposed in this research. In Section III-A, we describe the process of encoding a text into a numerical vector and the proposed similarity metric. In Section III-B, we describe the standard version of AHC algorithm where the proposed similarity metric is adopted. In Section III-C, we derive the three variants from the standard version. In Section III-D, we present the system architecture of the text clustering system.

A. Encoding Process

This section is concerned with the process of encoding a text into a numerical vector. Words are used as features

for encoding a text so, and some are selected from words in a corpus. The similarity between words is computed based on texts with their collocations and is used as the similarity between features. The similarities among features are considered for computing the semantic similarity between texts. In this section, we describe the process of encoding a text into a numerical vector and the process of computing the similarity between texts.

The process of encoding texts into numerical vectors is illustrated in Figure 1. The entire text collection is indexed into a list of words which are feature candidates. Some words are selected as the features by criteria among the feature candidates in the second step. The weights of the selected features in the text are computed as their relationships with the text, as elements of the numerical vector. Refer to [8] for studying the process and the selection criteria in detail.

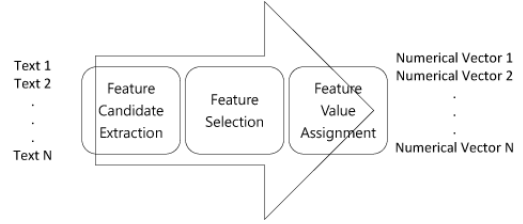


Figure 1. Process of Encoding Words into Numerical Vectors

The frame of computing the similarity between two numerical vectors is illustrated in Figure 2. The two d dimensional vectors and the d features are respectively notated respectively by $\mathbf{x} = [x_1 \ x_2 \ \dots \ x_d]$, $\mathbf{y} = [y_1 \ y_2 \ \dots \ y_d]$, and $f_1, f_2, \dots, f_d$. The feature similarity refers to the similarity between two features, which is notated by $\text{sim}(f_i, f_j)$. The feature value similarity is the similarity between two vectors, $\mathbf{x}$ and $\mathbf{y}$, such as cosine similarity. The similarity between words is computed by considering the two kinds of similarities.

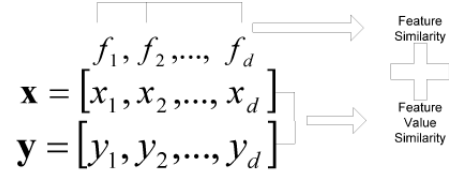


Figure 2. Frame of Computing Similarity between Numerical Vectors

Let us mention the process of computing the similarity between numerical vectors as the similarity between texts. The similarity between the features, f_i and f_j , is notated by $\text{sim}(f_i, f_j) = s_{ij}$. The similarity between the features is computed by equation (1),

$$s_{ij} = \text{sim}(t_i, t_j) = \frac{2 \times \#(t_i \wedge t_j)}{\#(t_i) + \#(t_j)}, \quad (1)$$

where $\#(t_i \wedge t_j)$, is the number of texts which include both words, t_i and t_j , $\#(t_i)$ is the number of texts which include the word, t_i , and $\#(t_j)$ is the number of texts which include the word, t_j , and the similarity matrix to the d features is constructed as a $d \times d$ square matrix as follows:

$$S = \begin{pmatrix} s_{11} & s_{12} & \dots & s_{1d} \\ s_{21} & s_{22} & \dots & s_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ s_{d1} & s_{d2} & \dots & s_{dd} \end{pmatrix}.$$

The similarity between the two numerical vectors, $\mathbf{x}$ and $\mathbf{y}$, is computed by equation (2),

$$\text{sim}(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^d \sum_{j=1}^d s_{ij} x_i y_j}{d \|\mathbf{x}\| \|\mathbf{y}\|} \quad (2)$$

Equation (2) is used for computing the similarity between a novice text and a sample text in the KNN algorithm.

Let us make some remarks on the encoding process and the computation of the similarity between two texts. A text is encoded into a numerical vector by the feature candidate extraction, the feature extraction, and the feature value assignment. The similarity between features which are given as words is computed by equation (1). The similarity between numerical vectors which represent texts is computed by equation (2). In future, we consider computing the similarities among some features rather than all features for improving the computation speed.

B. Feature Similarity based AHC Algorithm

This section is concerned with the initial version of AHC algorithm which considers the feature similarities. The version of AHC algorithm was initially proposed as the approach to the text clustering by Jo in 2018 [6]. The similarity metric between numerical vectors which was described in the previous section is used for computing the similarity between clusters. The AHC algorithm proceeds clustering data items with the bottom-up direction. This section is intended to describe the initial version of AHC algorithm from which its variants are derived.

The initial status for applying the AHC algorithm to the word clustering is illustrated in Figure 3. The words as clustering targets are encoded into numerical vectors by the process, which was described in Section III-A, and they are notated by a set, $Tr = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$. The clusters are defined as many as data items, as $C_1, C_2, \dots, C_N$, and each cluster includes a data item as $C_i = \{\mathbf{x}_i\}$. The initial status which is represented in Figure 00 is notated by $C_1 = \{\mathbf{x}_1\}, C_2 = \{\mathbf{x}_2\}, \dots, C_N = \{\mathbf{x}_N\}$. The cluster with only one item is called singleton, and singletons are constructed as many as data items in the initial status.

The process of computing the similarity between two clusters is illustrated in Figure 4. The two clusters are



Figure 3. Initial Clusters in using AHC Algorithm: Singletons

notated by two sets of items, $C_1 = \{\mathbf{x}_{11}, \mathbf{x}_{12}, \dots, \mathbf{x}_{1|C_1|}\}$ and $C_2 = \{\mathbf{x}_{21}, \mathbf{x}_{22}, \dots, \mathbf{x}_{2|C_2|}\}$. The similarity between clusters, C_1 and C_2 , is computed by equation (3),

$$\text{sim}(C_1, C_2) = \frac{1}{|C_1| \cdot |C_2|} \sum_{i=1}^{|C_1|} \sum_{j=1}^{|C_2|} \text{sim}(\mathbf{x}_{1i}, \mathbf{x}_{2j}). \quad (3)$$

If the similarity metric which was described in Section III-A is adopted, the similarity between two clusters is always a normalized value between zero and one, as expressed in equation (4),

$$0 \leq \text{sim}(C_1, C_2) \leq 1. \quad (4)$$

The complexity of computing the similarity between two clusters is expressed by $O(|C_1||C_2|)$.

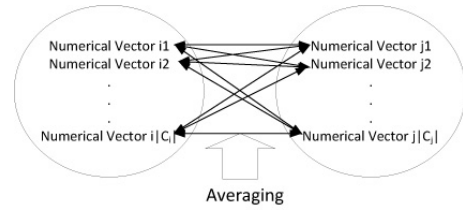


Figure 4. Similarity between Clusters

The process of clustering data items is illustrated as pseudo codes in Figure 5. The AHC algorithm begins with the status where singletons are given as many as data items. All possible pairs of clusters are generated from the current cluster list, their similarities are computed by equation (3), and the cluster pair with the highest similarity is merged into a cluster. In the AHC algorithm, the data items are clustered by iterating computing similarities of all possible cluster pairs and merge a cluster pair into a cluster. In each iteration, one cluster is decreased, and this iteration continues until reaching the desirable number of clusters.

Let us make some remarks on the initial version of the AHC algorithm from which we derive the variants. The initial status in applying the AHC algorithm to data clustering is that there are singletons as many as data items. The similarity between clusters is computed by averaging the similarities of all possible pairs from them. Data items are clustered by iterating computing the similarities of all possible cluster pairs and merge the pair with its maximal similarity into one cluster. In its variant, we will consider allow more than one pair of clusters to be merged.

```

List clusterDataItem(List numericalVectorList, int desiredClusterSize){
    int itemSize = numericalVectorList.size();

    if(itemSize <= desiredClusterNumber)
        return null;
    List clusterList = new List();
    //Initialize Clusters
    for(int i = 0; i < itemSize; i++){
        Cluster cc = new Cluster();
        NumericalVector dataitem = numericalVectorList.getElement(i);
        cc.addDataItem(dataitem);
        clusterList.addElement(cc);
    }

    int clusterListSize = clusterList.size();

    //Cluster Data Items in the bottom up direction
    while(clusterListSize > desiredClusterSize){
        double maxSimilarity = 0.0;
        int maxIndex1 = 0;
        int maxIndex2 = 0;
        for(int i = 0; i < clusterListSize; i++){
            Cluster c1 = clusterList.getElement(i);
            for(int j = 0; j < clusterListSize; j++){
                Cluster c2 = clusterList.getElement(j);
                double similarity = sim(c1, c2);
                if(similarity > maxSimilarity){
                    maxSimilarity = similarity;
                    maxIndex1 = i;
                    maxIndex2 = j;
                }
            }
        }
        Cluster cmax1 = clusterList.getElement(maxIndex1);
        Cluster cmax2 = clusterList.getElement(maxIndex2);
        Cluster mergeCluster = cmax1.merge(cmax2);
        clusterList.deleteElement(cmax1);
        clusterList.deleteElement(cmax2);
        clusterList.addElement(mergeCluster);
    }
}

```

Figure 5. Initial Version of AHC Algorithm

C. AHC Variants

This section is concerned with the variants which are derived from the AHC algorithm which was covered in Section III-B. The difference from the traditional version of AHC algorithm is to adopt the similarity metric which is expressed in equation (2) for computing the similarity between clusters. In this section, we derive the three variants by merging cluster pairs by adopt the threshold-based selection, allowing merge of multiple pairs in each iteration, and merging clusters within the radius. In this research, we adopt the three variants of AHC algorithm for implementing the text clustering system. This section is intended to describe the three variants of AHC algorithm.

In Figure 6, we illustrate the process of clustering data items by the first variant of AHC algorithm. The clusters are initialized with singletons like the initial version which was described in Section III-B. All possible cluster pairs are generated, and the cluster pairs whose similarities are more than or equality to the similarity threshold are merged. If there is no pair which satisfies the condition, clustering data items is terminated. In this variant, the number of clusters is decreased variably in this variant.

We illustrate the process of clustering data items by the second variant of AHC algorithm in Figure 7 as a pseudo code. The number of cluster pairs which are merged into each iteration is given as an additional parameter in this variant. All possible cluster pairs are generated, and the

```

List clusterDataItem(List numericalVectorList, double similarityThreshold){
    int itemSize = numericalVectorList.size();

    List clusterList = new List();
    //Initialize Clusters
    for(int i = 0; i < itemSize; i++){
        Cluster cc = new Cluster();
        NumericalVector dataitem = numericalVectorList.getElement(i);
        cc.addDataItem(dataitem);
        clusterList.addElement(cc);
    }

    boolean similarityFlag = false;

    //Cluster Data Items in the bottom up direction
    while(similarityFlag){
        int clusterListSize = clusterList.size();
        similarityFlag = false;
        for(int i = 0; i < clusterListSize; i++){
            Cluster c1 = clusterList.getElement(i);
            for(int j = 0; j < clusterListSize; j++){
                Cluster c2 = clusterList.getElement(j);
                double similarity = sim(c1, c2);
                if(similarity >= similarityThreshold){
                    Cluster mergeCluster = c1.merge(cmax2);
                    clusterList.updateElement(i, mergeCluster);
                    clusterList.deleteElement(j);
                    clusterListSize = clusterList.size();
                    similarityFlag = true;
                }
            }
        }
    }
}

```

Figure 6. Feature Similarity based AHC Variant 1: Similarity Threshold Based AHC Algorithm

similarity is computed for each pair. The cluster pairs are sorted by the descending order of the similarity, and the pairs within the rank are merged. The difference of this variant from the initial version is to merge multiple cluster pairs, instead of one pair.

```

List clusterDataItem(List numericalVectorList, int desiredClusterSize, int rank){
    int itemSize = numericalVectorList.size();

    List clusterList = new List();
    //Initialize Clusters
    for(int i = 0; i < itemSize; i++){
        Cluster cc = new Cluster();
        NumericalVector dataitem = numericalVectorList.getElement(i);
        cc.addDataItem(dataitem);
        clusterList.addElement(cc);
    }

    int clusterListSize = clusterList.size();

    //Cluster Data Items in the bottom up direction
    while(clusterListSize > desiredClusterSize){
        List pairList = new List();
        for(int i = 0; i < clusterListSize; i++){
            Cluster c1 = clusterList.getElement(i);
            for(int j = i; j < clusterListSize; j++){
                Cluster c2 = clusterList.getElement(j);
                Pair indivPair = new Pair(c1, c2);
                indivPair.computeSimilarity();
                pairList.addElement(indivPair);
            }
        }
        pairList.sortPairs();
        List selectedPairList = pairList.getSubList(0, rank-1);
        clusterList.mergeClusters(selectedPairList);
        clusterListSize = clusterList.size();
    }
}

```

Figure 7. Feature Similarity based AHC Variant 2: Merge of Multiple Pairs

In Figure 8, we illustrate the process of clustering data items by the last variant of AHC algorithm. The similarity threshold is given as an external parameter, and the number

of other clusters whose similarity is greater than or equality to the similarity threshold is counted. In each iteration, the cluster with its maximal number of its similar clusters is appointed as the pivot, and the pivot cluster and its similar ones are merged into one cluster. This variant iterates selecting the pivot cluster in the current cluster list and merge the pivot and its similar ones, as the process of clustering data items. If there is no cluster with its similar cluster, the iteration is terminated.

```

List clusterDataItem(List<numericalVector> list, double similarityThreshold){
    int itemSize = numericalVectorList.size();

    List<clusterDataItem> clusterList = new List<>();
    //Initialize Clusters
    for(int i = 0; i < itemSize; i++){
        Cluster cc = new Cluster();
        NumericalVector dataItem = numericalVectorList.getElement(i);
        cc.addDataItem(dataItem);
        clusterList.addElement(cc);
    }

    //Cluster Data Items in the bottom up direction
    while(true){
        int clusterListSize = clusterList.size();
        if(clusterListSize == 1)
            return;
        int maxSimilarClusterSize = 0;
        int maxIndex = 0;
        for(int i = 0; i < clusterListSize; i++){
            Cluster c1 = clusterList.getElement(i);
            int similarClusterSize = c1.countSimilarClusters(clusterList);
            if(similarClusterSize > maxSimilarClusterSize){
                maxSimilarClusterSize = similarClusterSize;
                maxIndex = i;
            }
        }
        if(maxSimilarClusterSize == 0)
            return;
        Cluster pivotCluster = clusterList.getElement(maxIndex);
        List<similarClusterList> = clusterList.getSimilarClusterList(pivotCluster);
        clusterList.mergeClusterList(similarClusterList);
    }
}

```

Figure 8. Feature Similarity based AHC Variant 3: Radius based AHC Algorithm

Let us make some remarks on the three variants of AHC algorithm which are adopted as the approaches to the word clustering. In each iteration of the first variant, cluster pairs with their similarities which are greater than or equal to the threshold are merged. In each iteration of the second variant, a fixed number of cluster pairs with their highest similarities are merged. In each iteration of the third variant, a pivot cluster and its similar clusters are merged. We may derive more variants from the AHC algorithm by setting different policies on merging clusters.

D. System Architecture

This section is concerned with the architecture and the execution flow of the text clustering system. In Section III-C, we describe the three AHC variants which consider the feature similarities and adopt them for implementing the text clustering system. In the architecture of the text clustering system, which was previous proposed, the initial AHC algorithm which is mentioned in Section III-B is replaced by its variants. The process of executing the system is to encode the texts into numerical vectors and cluster

them. This section is intended to describe the text clustering system which is implemented in this study with respect to its architecture.

A group texts and numerical vectors is illustrated in Figure 9. All of texts are initially given at a time under the assumption of the offline clustering. The texts in the group are encoded into numerical vectors by the process which is described in Section III-A. They are clustered by the AHC algorithm which is described in Section III-C. The online clustering where texts are given as a stream will be considered in next research.

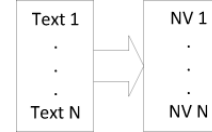


Figure 9. Preparation of Training Examples

The architecture of the text clustering system is illustrated in Figure 10. In the encoding module, texts as clustering targets are encoded into numerical vectors. The clustering module includes the similarity computation module which computes the similarities among the numerical vectors by the process which is described in Section III-A and clusters them by one of the AHC variants. The text clusters are generated as the final output from the system. The difference from the system architecture which was proposed in [6] is to replace the initial AHC algorithm by its variants.

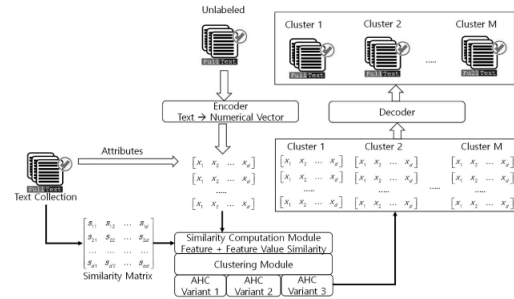


Figure 10. System Architecture

The execution flow of the text clustering system is illustrated in Figure 11. The texts are encoded into numerical vectors, and the similarity matrix is constructed from the text collection for computing the similarity. The similarity metric which considers both the feature similarity, and the feature value similarity is used for computing the similarity between clusters. The data items are clustered, selecting one among the three AHC variants which are described in Section III-C. The external parameter, the similarity threshold, is controlled in the first and the third variant for reaching the desired number of clusters.

Let us make some remarks on the proposed version of the text clustering system whose architecture is presented in

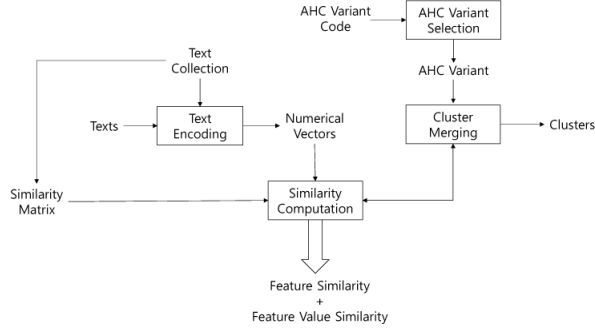


Figure 11. System Flow

Figure 10. The offline clustering is assumed in implementing the system; it will be expanded for doing online clustering in the next research. The upgrading point of the system is to replace the initial AHC algorithm by its variants which are described in Section III-C. The texts are encoded into numerical vectors, the similarity between clusters is computed by equation (3), and they are clustered by one of the AHC variants. It is expected that the clustering performance and speed are improved, by replacing the initial version by its variants.

IV. CONCLUSION

Let us mention the significances of this research as the conclusion. We derive the three variants from the standard version by allowing multiple clusters to be merged at a time. We encode words into numerical vectors and apply the similarity metric among them to the AHC variants as well as the standard version. We design the text clustering system by adopt the AHC variants with the proposed similarity metric. In the next research, we will implement the text clustering system as a real system.

Let us mention the remaining tasks as the further discussions on this research. We need to solve the high complexity in computing the proposed similarity metric by allowing only some features to compute the similarities, rather than the entire attributes. We may modify other machine learning algorithms such as the Naïve Bayes and the SVM (Support Vector Machine) by applying the proposed similarity metric. We may apply the proposed versions of the AHC variants to other clustering tasks. The text clustering may be implemented by adopting the proposed AHC variants as real programs.

REFERENCES

[1] P. Mitra, C. A. Murthy, and S. K. Pal. "Unsupervised feature selection using feature similarity", 301-312, IEEE transactions on pattern analysis and machine intelligence 24.3 2002.

[2] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: an overview", 86-97, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2, 1, 2012.

[3] K. Sasirekha and P. Baby. "Agglomerative hierarchical clustering algorithm-A Review", 83-85 International Journal of Scientific and Research Publications, 3,1, 2013.

[4] Z. Nazari, D. Kang, M.R. Asharif, Y. Sung, and S. Ogawa, "A new hierarchical clustering algorithm", 148-152, The Proceedings of IEEE International Conference on Intelligent Informatics and Biomedical Sciences, 2015.

[5] T. Jo, "AHC Algorithm for Text Clustering using Attribute Similarity", 148-152, The Proceedings of International Conference on Green and Human Information Technology, 2018.

[6] T. Jo, "Clustering Texts using Feature Similarity based AHC Algorithm", pp5993-6003, Journal of Intelligent and Fuzzy Systems, Vol 35, 2018.

[7] T. Jo, "Semantic Word Categorization using Feature Similarity based K Nearest Neighbor", 67-78, Journal of Multimedia Information Systems, 2018.

[8] T. Jo, "Text Mining: Concepts and Big Data Challenge", Springer, 2019.

[9] T. Jo, "Text Classification using Feature Similarity based K Nearest Neighbor", 13-21, AS Medical Science, 3, 4, 2019.

[10] Taeho Jo, "Content based Word Clustering using Feature Similarity based AHC Algorithm", in Preparation, 2023.